

## データベース操作装置およびその方法

## 技術分野

- 5           本発明は、データベースなどに対する接続を制御する接続制御装置およびその方法に関する。

## 背景技術

- 10           データを保存し、多くのユーザの間で共用するために、情報処理の様々な分野において、データベースが利用されている。

データベースの操作のためには、IBM社により開発されたSQL (Structured Query Language) が、現在、一般的に用いられている。

SQL記述されたプログラムを解析し、データの記憶・検索などを行うシステムとして、DBMS (Data Base Management System) が知られている。

- 15           SQLおよびDBMSの応用製品は、<http://www.microsoft.com/sql/>など、様々な文献に開示されている。

また、例えば、<http://www.microsoft.com/japan/com/compres.asp>は、マイクロソフト社により開発され、データベースの操作をサポートするCOM+を開示する。

- 20           データベースの操作は、現在、一般的に、その処理を、ユーザインターフェースを提供するUI層と、ビジネスのための機能などを提供するアプリケーション層と、データベースへのアクセス機能を提供するデータ層とに分けて行うことにより実現されている。

- 25           このように、その操作を、3つの階層に機能分担させて実現するデータベースは、3階層システムなどとも呼ばれる。

しかしながら、この3階層データベースにおいては、アプリケーション層のプログラミングが煩雑になりやすく、その工数も多く必要とされる傾向にある。

上述したCOM+は、3階層データベースに適合され、各階層の独立性

を高めて、そのプログラミングを容易にする。

しかしながら、COM+を用いると、各階層の独立性が高くなるので、却って、作成されたプログラムの全階層を通じたデバッグが難しくなる傾向があり、従って、製品出荷後のメンテナンスも難しくなる傾向がある。

5

[非特許文献 1] <http://www.microsoft.com/sql/>

[非特許文献 2] <http://www.microsoft.com/japan/com/compres.asp>

### 発明の開示

10           本発明は、上述のような背景からなされたものであり、操作のためのプログラミングが容易なデータベース操作装置およびその方法を提供することを目的とする。

また、本発明は、操作のためのプログラムのデバッグおよびメンテナンスが容易なデータベース操作装置およびその方法を提供することを目的とする。

15           上記目的を達成するために、本発明にかかるデータベース操作装置は、複数の階層に分かれた処理により、データベースを操作するデータベース操作装置であって、前記処理は、他の階層の処理を起動する起動処理、および、前記起動処理により起動される被起動処理またはこれらのいずれかであって、前記被起動処理の 1 つ以上は、前記データベースに対する操作を行うデータベース操作処理  
20           であって、1 つ以上の前記起動処理を含む処理集合を定義する集合定義手段と、前記処理集合に含まれる起動処理の進行と、前記処理集合に含まれる起動処理により起動されたデータベース操作処理の処理結果とに基づいて、少なくとも、このデータベース操作処理による前記データベースに対する操作の内容を制御する処理制御手段とを有する。

25           好適には、前記階層は、3 つ以上に分かれ、前記処理制御手段は、前記処理集合に含まれる起動処理の進行と、前記処理集合に含まれる起動処理により起動されたデータベース操作処理以外の被起動処理の処理結果とに応じて、前記データベース操作処理以外の被起動処理の処理内容を、さらに制御する。

好適には、1 つ以上の前記データベース操作処理を含むライブラリ手段

をさらに有し、前記データベース操作処理を起動する起動処理は、前記ライブラリ手段に含まれる前記データベース操作処理を起動する。

好適には、前記処理集合それぞれに対して、この処理集合に含まれる起動処理、および、この処理集合に含まれる起動処理により起動される被起動処理のための記憶領域を設定する記憶領域設定手段と、前記処理集合それぞれに含まれる処理のために用いられるデータを、前記処理集合それぞれに対して設定された記憶領域において管理するデータ管理手段とをさらに有する。

好適には、前記被起動処理それぞれは、この被起動処理を起動した起動処理に対して、処理の結果を示す戻り値を返し、前記階層は、インターフェース層と、アプリケーション層と、データベース層とを含み、前記インターフェース層は、前記起動処理として、外部からの操作に応じて、前記アプリケーション層に含まれる被起動処理を起動し、起動した被起動処理から返された戻り値に応じた処理を行うユーザインターフェース処理、1つ以上を含み、前記アプリケーション層は、前記起動処理および前記被起動処理として、前記インターフェース処理により起動され、前記データベース層に含まれるデータベース操作処理を1つ以上、起動し、前記起動されたデータベース操作処理からの戻り値に基づいて、前記データベースを用いたサービスを実現し、前記戻り値として、このサービスの結果を、前記ユーザインターフェース処理に返すアプリケーション処理、1つ以上を含み、データベース層は、前記データベース操作処理として、前記アプリケーション処理により起動され、前記データベースに対する操作を行い、前記戻り値として、このデータベースに対する操作の結果を、前記アプリケーション処理に返すデータベース操作処理、1つ以上を含む。

好適には、前記処理制御手段は、前記処理集合に含まれる前記アプリケーション処理により、前記データベース操作処理が、最初に起動されたときに、前記起動されたデータベース操作処理と、前記データベースとを接続し、前記処理集合に含まれる前記アプリケーション処理により、最後に起動された前記データベース操作処理が終了したとき、または、前記データベース操作処理が失敗したときに、前記起動されたデータベース操作処理と、前記データベースとを切断するように、前記データベース操作処理の実行を制御する。

好適には、前記処理制御手段は、前記処理集合に含まれる前記アプリケーション処理により、最後に起動された前記データベース操作処理が成功したときに、前記処理集合に含まれる前記アプリケーション処理により起動された前記データベース操作処理の結果を、前記データベースにおいて確定させ、これ以外  
5 のときには、前記データベースを、前記処理集合に含まれる前記アプリケーション処理により、前記データベース操作処理が最初に起動される前の状態に戻す。

好適には、前記処理制御手段は、前記処理集合に含まれる前記アプリケーション処理により起動された前記データベース操作処理が失敗したときに、前記データベースを、前記処理集合に含まれる前記アプリケーション処理により、  
10 前記データベース操作処理が最初に起動される前の状態に戻す。

また、本発明にかかる情報処理装置は、複数の階層に分かれた処理により、所定の情報処理を行う情報処理装置であって、前記処理は、他の階層の処理を起動する起動処理、および、前記起動処理により起動される被起動処理またはこれらのいずれかであって、1つ以上の前記起動処理を含む処理集合を定義する  
15 集合定義手段と、前記処理集合に含まれる起動処理の進行と、前記処理集合に含まれる起動処理により起動された被起動処理の処理結果とに応じて、この被起動処理の処理内容を制御する処理内容制御手段とを有する。

また、本発明にかかるデータベース操作方法は、複数の階層に分かれた処理により、データベースを操作するデータベース操作方法であって、前記処理  
20 は、他の階層の処理を起動する起動処理、および、前記起動処理により起動される被起動処理またはこれらのいずれかであって、前記被起動処理の1つ以上は、前記データベースに対する操作を行うデータベース操作処理であって、1つ以上の前記起動処理を含む処理集合を定義する定義ステップと、前記処理集合に含まれる起動処理の進行と、前記処理集合に含まれる起動処理により起動されたデータベース操作処理の処理結果とに基づいて、少なくとも、このデータベース操作  
25 処理による前記データベースに対する操作の内容を制御する処理制御ステップとを含む。

また、本発明にかかる情報処理方法は、複数の階層に分かれた処理により、所定の情報処理を行う情報処理方法であって、前記処理は、他の階層の処理

を起動する起動処理、および、前記起動処理により起動される被起動処理またはこれらのいずれかであって、1つ以上の前記起動処理を含む処理集合を定義する集合定義ステップと、前記処理集合に含まれる起動処理の進行と、前記処理集合に含まれる起動処理により起動された被起動処理の処理結果とに応じて、この被

5 起動処理の処理内容を制御する処理内容制御ステップとを含む。

また、本発明にかかる第1のプログラムは、複数の階層に分かれた処理により、データベースを操作するデータベース操作装置のプログラムであって、前記処理は、他の階層の処理を起動する起動処理、および、前記起動処理により起動される被起動処理またはこれらのいずれかであって、前記被起動処理の1つ

10 以上は、前記データベースに対する操作を行うデータベース操作処理であって、1つ以上の前記起動処理を含む処理集合を定義する集合定義ステップと、前記処理集合に含まれる起動処理の進行と、前記処理集合に含まれる起動処理により起動されたデータベース操作処理の処理結果とに基づいて、少なくとも、このデータベース操作処理による前記データベースに対する操作の内容を制御する処理制

15 御ステップとをコンピュータに実行させる。

また、本発明にかかる第2のプログラムは、複数の階層に分かれた処理により、所定の情報処理を行う情報処理装置のプログラムであって、前記処理は、他の階層の処理を起動する起動処理、および、前記起動処理により起動される被起動処理またはこれらのいずれかであって、1つ以上の前記起動処理を含む処

20 理集合を定義する集合定義ステップと、前記処理集合に含まれる起動処理の進行と、前記処理集合に含まれる起動処理により起動された被起動処理の処理結果とに応じて、この被起動処理の処理内容を制御する処理内容制御ステップとをコンピュータに実行させる。

本発明によれば、操作のためのプログラミングが容易なデータベース操作装置およびその方法が提供される。

25

また、本発明によれば、操作のためのプログラムのデバッグおよびメンテナンスが容易なデータベース操作装置およびその方法が提供される。

図面の簡単な説明

図 1 は、S Q L (Structured Query Language) により記述されたデータベース操作のプログラムの第 1 の例を、リスト形式で示す図である。

図 2 は、S Q L により記述されたデータベース操作のプログラムの第 2 の例を、リスト形式で示す図である。

5 図 3 は、図 2 に示したデータベース操作のプログラムの処理 (S 1 0) を示すフローチャートである。

図 4 は、図 2 に示したデータベース操作のプログラムの処理を、概念的に示す図である。

10 図 5 は、S Q L により記述されたデータベース操作のプログラムの第 3 の例を、リスト形式で示す図である。

図 6 は、図 5 に例示したプログラムに対するデバッグ作業の困難さを説明する図である。

図 7 は、本発明にかかるデータベース操作方法が適用されるネットワークシステムの構成を例示する図である。

15 図 8 は、図 7 に示したサーバ、開発・操作 P C およびクライアント P C のハードウェア構成を例示する図である。

図 9 は、図 7 に示したサーバにおいて実行されるサーバプログラムの構成を示す図である。

20 図 1 0 は、図 7 に示した開発・操作 P C において実行される開発・制御プログラムの構成を示す図である。

図 1 1 は、図 1 0 に示したプログラム開発支援部などにより作成される制御プログラムの例を、リスト形式で示す図である。

図 1 2 は、図 1 1 に示した制御プログラムの処理 (S 1 2) を示すフローチャートである。

25 図 1 3 は、図 1 1 に示した制御プログラムの処理およびその実行管理を、概念的に示す図である。

図 1 4 は、図 1 0 に示した開発・制御プログラムのプログラム実行制御部による制御プログラム (図 1 1 ~ 図 1 3) の処理の実行の制御 (S 2 0) を示すフローチャートである。

図 1 5 は、開発・制御用プログラム（図 1 0）による制御プログラムに対するデバッグの容易さを説明する図である。

図 1 6 は、各階層に複数のトランザクションを含む制御プログラムを例示する図である。

5

#### 発明を実施するための最良の形態

##### 〔本発明がなされるに至った背景〕

本発明の理解を容易にするために、まず、本発明がなされるに至った背景を、さらに説明する。

10 一般に、データベースの操作のためには、ユーザからの操作を受け入れ、処理結果をユーザに表示するユーザインターフェース機能と、ユーザの操作に応じて、データベースを操作し、様々なサービスをユーザに提供するアプリケーション機能と、実際にデータベース（DBMS）に対するアクセスを行うデータベース機能とが必要とされる。

15 図 1 は、SQL (Structured Query Language) により記述されたデータベース操作のプログラムの第 1 の例を、リスト形式で示す図である

例えば、図 1 にリストとして示すように、これら 3 つの機能を階層分けせずに実現することも可能である。

つまり、図 1 に示すプログラムは、以下のように、データベースに対する操作を行うように記述されている。

20

まず、ユーザによる操作があるたびに、まず、データベース（DB）への接続が行われ、トランザクションが開始される。

次に、トランザクションが開始され、さらに、データベースに対する必要な操作が、順次、行われる。

25 データベースに対する操作が終了すると、最後に、トランザクションにおいて、データベースに生じた変更内容などが確定（コミット）され、データベースへの接続が終了される。

しかしながら、階層分けせずにプログラムを記述すると、ユーザによる操作および提供されるサービスの組み合わせの全てについて、図 1 に例示したよ

うなプログラムを作成しなければならない。

従って、階層分けせずにプログラムを記述すると、プログラムの量が多くなる上に、システムとしての構成が柔軟性を欠き、また、プログラムモジュール（関数）のコンポーネント化が難しいので、機能の変更・追加などがしにくい

5

このような理由から、現在では、データベースの操作のためのプログラムは、ユーザインターフェースを実現するユーザインターフェース層（UI層）、データベースを応用した様々なサービスを提供するアプリケーション層（AP層）、および、実際にデータベースに対する操作を行うデータベース層（DB層）に分けられた形式で作成されるようになってきている。

10

図2は、SQLにより記述されたデータベース操作のプログラムの第2の例を、リスト形式で示す図である

図3は、図2に示したデータベース操作のプログラムの処理（S10）を示すフローチャートである。

15

図4は、図2に示したデータベース操作のプログラムの処理を、概念的に示す図である。

図2～図4に示すように、ユーザインターフェース層のプログラムの処理において、コンピュータなどの画面に表示されたユーザインターフェース画像内（図示せず）の「ボタン5 (button5)」に対するユーザによるクリックが検出されると、アプリケーション層のプログラムが起動される（S100，S102；図3）。

20

アプリケーション層の処理において、まず、データベースへの接続が行われ、データベースがオープンされて、トランザクションが開始される（S104；図3）。

25

次に、サービスの提供のために、データベースに対する操作（DB操作#1，#2）が、順次、実行され、これらの処理が実行されるたびに、データベース層のプログラムが起動される（S106，S108；図3）。

アプリケーション層の処理は、全てのデータベースに対する操作が成功したときには、データベースにおける操作結果を確定（コミット）し、データベ

ースへの接続を終了（クローズ）し、処理結果および戻り値を、ユーザインターフェース層の処理に返す（S 1 1 0, S 1 1 2 ; 図 3）。

アプリケーション層の処理は、これ以外ときには、データベースに対する操作を取り消して内容を復元（ロールバック）する（S 1 1 4 ; 図 3）。

- 5           最後に、アプリケーション層の処理は、データベースへの接続を終了（クローズ）し、処理結果および戻り値を、ユーザインターフェース層の処理に返す（S 1 1 6 ; 図 3）。

ユーザインターフェース層の処理により、アプリケーション層の処理結果がユーザに示される。

- 10           図 2 に例示したように、データベースに対する操作を、ユーザインターフェース層と、アプリケーション層と、データベース層とに分けてプログラミングすると、プログラムモジュールのコンポーネント化が容易になるので、図 1 に例示したように階層分けせずにプログラミングする場合に比べて、機能の変更・追加などが容易になる。

- 15           しかしながら、図 2 に例示した場合には、データベースへの接続と、トランザクションの管理とを共通とするためには、アプリケーション層の処理において、データベースへの接続のためのオブジェクト（con）、および、トランザクションのためのオブジェクト（txのこと）を作成しなければならない。

- 20           作成されたこれら 2 つのオブジェクトは、引数として、データベース層の処理を実現するプログラムモジュールの呼び出しごとに、毎回、渡されることとなる。

- 25           また、これら 2 つのオブジェクトは、データベース層において、データベースを実際に操作する処理のプログラムモジュールに対してばかりではなく、データベースを操作しない処理のプログラムモジュールに対しても、引数として渡さなければならないので、アプリケーション層のプログラミングが煩雑になる傾向がある（ただし、ここで述べたデータベースを操作しない処理のプログラムモジュールは、図 2 に示したリストには含まれない。アプリケーション層にサブルーチンを作成し、その中からデータベース層を呼ぶときに、そのサブルーチンでは con, tx を使わないにもかかわらず、下位のデータベース層に引き渡すた

めだけに、引数を二つ付加しなければならない。このような引数の付加は、これはサブルーチンが増えると、かなりの手間となる）。

引数を渡さなくてもよいように、データベース層の処理を実現するプログラムモジュールを作成することも可能であるが、この場合には、プログラムモジュールの機能を引数によって変更することができないので、データベース層に  
5     、機能が少しずつ異なるプログラムモジュールを、数多く作成する必要が生じる。

従って、この場合には、データベース層におけるプログラミングの工数が増えたり、作成されたプログラムのデバッグおよびメンテナンスのために、多  
10    くの手間が必要となったりする。

データベース層のプログラムモジュールをコンポーネント化するために、マイクロソフト社のオペレーティングシステムWindows（登録商標）には、COM+と呼ばれる機能が付加されている。

図5は、SQLにより記述されたデータベース操作のプログラムの第  
15    3の例を、リスト形式で示す図である

COM+を利用して、図2に示したものと同様のプログラムを作成すると、図5に例示する通りとなる。

図5に例示するように、ユーザインターフェース層のプログラムの処理において、「ボタン5」に対するクリックが検出されると、アプリケーション層  
20    のプログラムが起動される。

アプリケーション層の処理において、まず、トランザクションの動作を指定する属性が指定され、サービスの提供のために、データベースに対する操作（DB操作#1、#2）が、順次、実行され、これらの処理が実行されるたびに、データベース層のプログラムが起動される。

25    起動されたデータベース層の処理結果は、戻り値としてアプリケーション層に返され、アプリケーション層の処理結果および戻り値は、ユーザインターフェース層の処理に返される。

ユーザインターフェース層の処理により、アプリケーション層の処理結果がユーザに示される。

このように、COM+を利用すると、アプリケーション層に対するデータベース層のプログラムモジュールの独立性が高まる。

また、アプリケーション層から、データベース層のプログラムモジュールそれぞれに、引数を渡す必要がなくなるので、プログラミングの内容がシンプルになる。

しかしながら、COM+を利用したプログラミングを行うプログラマは、当然ながら、COM+特有の仕様を学習し、その利用法に習熟しなければならないので、そのために、多くの手間と時間が必要となる。

図6は、図5に例示したプログラムに対するデバッグ作業の困難さを説明する図である。

また、図5に例示したユーザインターフェース層の処理は、一般的なSQL処理方式に従って実行されるのに対し、アプリケーション層およびデータベース層の処理は、COM+特有の処理方式に従って実行される。

図6に示すユーザインターフェース（UI）層の処理を実現するプロセスが動作しており、デバッガをアタッチして、デバッグを行うことができたとしても、アプリケーション層およびデータベース層のプロセスは、これらが起動されない限り存在しないことがあるので、デバッガをアタッチすることができるとは限らない。

従って、ユーザインターフェース層、アプリケーション層およびデータベース層を通したプログラムの動作確認およびデバッグを行うためには、例えば、ユーザインターフェース層のプロセスにおいて、アプリケーション層のプログラムを起動する処理にブレークポイントを設定し、その実行を停止してから、起動されたアプリケーション層のプロセスに対して、別途、デバッガをアタッチし、デバッグを行うようにしなければならない。

従って、この場合には、COM+を採用しない場合に比べても、プログラムのデバッグおよびメンテナンスに、却って多くの手間と時間とが必要となりかねない。

本発明は、以上説明した背景からなされたものであって、本発明の実施形態によると、ここまでに背景として述べた方式の不利な点が全て解消される。

なお、以下に説明する本発明の実施形態は、例示であって、本発明の技術範囲の限定を意図するものではない。

また、以下に説明する本発明の実施形態の構成部分における機能分担は、固定的ではなく、1つの構成部分の機能が、複数の構成部分に分散されたり、  
5 複数の構成部分の機能が、1つの構成部分に集約されたりする。

#### 〔実施形態〕

以下、本発明の実施形態を説明する。

図7は、本発明にかかるデータベース操作方法が適用されるネットワークシステム1の構成を例示する図である。  
10

図7に示すように、ネットワークシステム1は、サーバシステム2およびクライアントコンピュータ（クライアントPC）102-1～102-nとが、LAN、WANあるいはインターネットなどのネットワーク100を介して接続された構成を採る。

15 サーバシステム2は、サーバ3と開発・操作用PC4とが、LAN20を介して接続された構成を採る。

ネットワークシステム1において、サーバシステム2は、例えば、データベースを利用したWebサーバとしての機能を有し、クライアントPC102-1～102-nに対して、様々なサービスを提供する。

20 なお、以下、クライアントPC102-1～102-nなど、複数ある構成部分は、単にクライアントPC102などと略記されることがある。

また、ネットワークシステム1の各構成部分を、ノードと記すことがある。

図8は、図7に示したサーバ3、開発・操作用PC4およびクライアントPC102のハードウェア構成を例示する図である。  
25

図8に示すように、サーバ3、開発・操作用PC4およびクライアントPC102は、CPU122、メモリ124およびこれらの周辺回路などを含むコンピュータ本体120、表示装置、キーボードおよびマウスなどを含む入力・出力装置126、FD、CDおよびHDなどの記録媒体130に対してデータを

記録・再生する記録装置 128、および、ネットワーク 100（図 7）を介して他のノードと通信を行う通信装置 132 などから構成される。

つまり、サーバ 3、開発・操作用 PC 4 およびクライアント PC 102 は、ネットワーク 100 あるいは LAN 20 を介して、他のノードとの間で通信を行うことができる一般的なコンピュータとしての構成部分を有する。

図 9 は、図 7 に示したサーバ 3 において実行されるサーバプログラム 30 の構成を示す図である。

図 9 に示すように、サーバプログラム 30 は、データベース部 300、データベース管理システム部（DBMS）302 およびサービス提供部 304 から構成される。

サーバプログラム 30 は、例えば、記録媒体 130 を介してサーバ 3 に供給され、メモリ 124（図 8）にロードされて実行される（以下のプログラムについて同様）。

サーバプログラム 30 は、これらの構成部分により、開発・操作用 PC 4 の制御にしたって、クライアント PC 102（図 7）に対して、様々なサービスを提供する。

サーバプログラム 30 において、DBMS 302 は、開発・操作用 PC 4 からの制御に従って、データベース部 300 を操作して、サービス提供部 304 から入力されるデータをデータベース部 300 に記憶する。

また、DBMS 302 は、データベース部 300 を操作して、データベース部 300 に記憶したデータを読み出し、サービス提供部 304 に対して出力する。

データベース部 300 は、DBMS 302 による操作に従って、DBMS 302 から入力されるデータを記憶し、また、記憶したデータを DBMS 302 に対して出力する。

つまり、データベース部 300 および DBMS 302 は、上述したデータベース層の処理を行い、サービス提供部 304 に対して、データベースの機能を提供する。

サービス提供部 304 は、開発・操作用 PC 4 からの制御に従って、ク

クライアントPC 102などに対するサービス、例えば、データベースサーバあるいはWebサーバとしてのサービスを実現する。

つまり、サービス提供部304は、上述したユーザインターフェース層の処理およびアプリケーション層の処理を行い、クライアントPC 102に表示された操作画像（図示せず）に対するユーザの操作に応じて、データベース部300およびDBMS 302により提供されるデータベース機能を利用し、クライアントPC 102に対して、データベース部300に記憶されたデータを提供し、あるいは、クライアントPC 102から入力されるデータを、データベース部300に記憶させる。

10       なお、サービス提供部304は、開発・操作用PC 4におけるデバッグ作業などのために、開発・操作用PC 4から入力されたデータをデータベース部300に記憶させ、また、データベース部300に記憶されたデータを開発・操作用PC 4に対して出力することもある。

15       図10は、図7に示した開発・操作用PC 4において実行される開発・制御用プログラム40の構成を示す図である。

図10に示すように、開発・制御用プログラム40は、UI部400、プログラム開発部42およびプログラム実行制御部46から構成される。

プログラム開発部42は、プログラム開発支援部420およびデバッガ422などを含む。

20       プログラム実行制御部46は、実行制御部48、UI層制御部460、AP層制御部462およびDB層制御部464などを含み、実行制御部48は、UI・AP層実行管理部480、DB層実行管理部482、データ管理部484、メモリ領域管理部486、ライブラリ488およびトランザクション管理部490から構成される。

25       開発・制御用プログラム40は、これらの構成部分により、ユーザの操作に応じて、サーバプログラム30を制御する制御プログラム44の開発およびそのデバッグを行う。

また、開発・制御用プログラム40は、制御プログラム44の実行を制御して、サーバプログラム30におけるデータベース操作、および、データベ

スを利用した所望のサービスを実現する。

開発・制御用プログラム40において、UI部400は、開発・操作用PC4のユーザに対するユーザインターフェース機能を提供する。

また、UI部400は、ユーザの操作に応じて、開発・制御用プログラム40の各構成部分の処理を制御する。

プログラム開発部42において、プログラム開発支援部420は、ユーザ（プログラマ）によるSQL形式の制御プログラム44の開発を支援する。

プログラム開発支援部420により開発された制御プログラム44の例は、図11～図13を参照して後述される。

デバッガ422は、プログラム開発支援部420などを用いて開発された制御プログラム44のステップ実行など、制御プログラム44に対するデバッグ機能を提供する。

図11は、図10に示したプログラム開発支援部420などにより作成される制御プログラム44の例を、リスト形式で示す図である。

図12は、図11に示した制御プログラム44の処理（S12）を示すフローチャートである。

図13は、図11に示した制御プログラム44の処理およびその実行管理を、概念的に示す図である。

プログラム開発支援部420により、図11～図13に示すような制御プログラム44が作成される。

なお、この制御プログラム44の処理の内容は、比較しやすいように、図1，図2，図5に例示したプログラムと同様になっている。

図11～図13に示すように、制御プログラム44のユーザインターフェース層の処理（S120）において、クライアントPC102（図8）の入力・出力装置126の表示装置などの画面に表示されたユーザインターフェース画像内（図示せず）の「ボタン5 (button5)」に対するユーザによるクリックが検出されると、アプリケーション層のプログラムが起動される（S120；図12）。

起動されたアプリケーション層の処理（S122）において、まず、ト

ランザクションの単位が指定される。

このトランザクションは、アプリケーション層の処理（S 1 2 2）において、サーバ 3 によるサービスの提供のために必要な 1 つ以上のデータベース層の処理の起動を含む。

- 5           なお、図 1 1 ～図 1 3 においては、DB 操作 # 1, # 2 のために、データベース層の処理が起動される場合が例示されている。

トランザクションの定義とともに、このトランザクションにおける処理のために、専用の連続したメモリ領域が、開発・操作用 PC 4 のメモリ 1 2 4（図 8）上に確保される（図 1 3）。

- 10           トランザクションそれぞれに含まれる全ての処理において用いられるデータは、トランザクションそれぞれのために専用に確保されたメモリ領域に記憶され、管理される。

このように確保されたメモリ領域には、データベースに対する接続に必要な接続オブジェクトも記憶される。

- 15           記憶された接続オブジェクトは、必要に応じて処理のために用いられるので、引数の受け渡しが不要となる。

次に、サーバ 3 に対する操作（DB 操作 # 1, # 2）のために、データベース層の処理が、順次、起動される。

これらの操作が終了すると、アプリケーション層の処理が終了する。

- 20           データベース層の処理（S 1 2 4）において、データベース（DBMS 3 0 2 およびデータベース部 3 0 0；図 9）に対する接続が行われる。

さらに、アプリケーション層の処理により起動されたデータベース操作のための処理が実行される。

- 25           データベース操作のための処理を行うプログラムモジュールそれぞれは、ライブラリに含まれるコンポーネント化されたプログラムモジュールとして提供される。

図 1 1 においては、このようにコンポーネント化されたデータベース層のプログラムモジュールとして、アプリケーション層において定義されたトランザクションにおいて、データベース（DBMS 3 0 2 およびデータベース部 3 0 0；図 9

）に接続するための"MyConnection"、および、このトランザクションにおけるデータベースの操作を行うための"MyTransaction"が例示されている。

データベース操作のための処理結果は、データベース部 300 において  
5 確定（コミット）され、あるいは、取り消されて復元（ロールバック）される。

これらの処理が終了すると、データベース層の処理が終了する。

図 11～図 13 に例示した制御プログラム 44 の処理の実行は、後述するように、プログラム実行制御部 46（図 10）により制御される。

再び図 10 を参照する。

10 実行制御部 48 において、UI・AP 層実行管理部 480 は、制御プログラム 44 のユーザインターフェース層のプログラムおよびアプリケーション層のプログラムを解釈し、トランザクションごとに、これらのプログラムの処理の実行を管理する。

DB 層実行管理部 482 は、制御プログラム 44 のデータベース層のプログラムを解釈し、トランザクションごとに、その処理の実行を管理する。  
15

メモリ領域管理部 486 は、トランザクションそれぞれの処理に用いられるメモリ領域（図 13）を、開発・操作用 PC 4 のメモリ 124（図 8）上に確保し、管理する。

また、メモリ領域管理部 486 は、処理が終了したトランザクションの  
20 メモリ領域を開放する。

データ管理部 484 は、トランザクションそれぞれの処理において、メモリ領域に記憶されるデータを管理する。

ライブラリ 488 は、データベース層の処理を行うコンポーネント化されたプログラムモジュールを提供する。

25 トランザクション管理部 490 は、アプリケーション層において定義されたトランザクションを管理する。

UI 層制御部 460 および AP 層制御部 462 は、UI・AP 層実行管理部 480 による管理に従って、サーバプログラム 30（図 9）のサービス提供部 304 の処理を制御する。

DB層制御部464は、DB層実行管理部482の管理に従って、DBMS302およびデータベース部300の処理を制御する。

図14を参照して、開発・制御用プログラム40のプログラム実行制御部46の処理を、さらに説明する。

- 5           図14は、図10に示した開発・制御用プログラム40のプログラム実行制御部46による制御プログラム44（図11～図13）の処理の実行の制御（S20）を示すフローチャートである。

なお、図14においては、図3に示した処理と対応する処理には、同じ符号が付されている。

- 10           図14に示すように、ステップ100（S100）において、サーバプログラム30のサービス提供部304（図8）は、ユーザによるクライアントPC102（図7）の入力・出力装置126に表示されたユーザインターフェース画像のボタン5の押下を検出する。

- サービス提供部304は、ボタン5の押下の検出結果を、実行制御部4  
15   8（図10）のUI層制御部460を介して、UI・AP層実行管理部480に対して出力する。

UI・AP層実行管理部480は、このボタン5の押下の検出結果を受け入れる。

- ステップ102（S202）において、UI・AP層実行管理部480  
20   （図10）は、制御プログラム44のユーザインターフェース層のプログラム（図11～図13）を解釈して実行し、ボタン5の押下が検出されたか否かを判断する。

プログラム実行制御部46は、ボタン5の押下が検出されたときにはS200の処理に進み、これ以外のときにはS200の処理に戻る。

- 25   ステップ200（S200）において、UI・AP層実行管理部480（図10）は、アプリケーション層のプログラムを解釈して実行し、トランザクションの属性を指定し、トランザクションを定義する。

トランザクションの指定は、アプリケーション層のラス宣言の部分でおこなわれ、図11には、マイクロソフト社のプログラミング言語C#を用いた場

合のトランザクションの指定が例示されている。

サーバ 3 に対する接続処理が行われるときには、この部分が参照され、その動作が変更される。

メモリ領域管理部 486 は、定義されたトランザクションの処理のため  
5 に用いられるメモリ領域を確保し、管理する。

データ管理部 484 は、確保されたメモリ領域に記憶されるデータを管理する。

トランザクション管理部 490 は、定義されたトランザクションを管理する。

10       ステップ 202 (S202) において、UI・AP 層実行管理部 480 (図 10) は、データベース (DBMS 302 およびデータベース部 300 ; 図 9) に対する操作のための第 1 の処理 (DB 操作 # 1) を開始し、データベース層のプログラムの処理を起動する。

第 1 の処理のためのデータベース層のプログラムは、ライブラリ 488  
15 により、コンポーネント化されたプログラムモジュールとして提供される。

起動されたデータベース層のプログラムは、DB 層実行管理部 482 により解釈され、DB 層実行管理部 482 は、データベースに対する接続を行い、データベースをオープンする。

20       ステップ 204 (S204) において、DB 層実行管理部 482 (図 10) は、ライブラリ 488 により提供されたデータベース操作 (DB 操作 # 1) のためのプログラムモジュールを解釈し、DB 層制御部 464 を介して、データベース (DBMS 302 およびデータベース部 300 ; 図 9) に対する操作を行う。

25       ステップ 206 (S206) において、DB 層実行管理部 482 は、データベースからの戻り値を判断し、S204 において実行された処理が成功したか否かを判断する。

プログラム実行制御部 46 は、処理が成功したときには S208 の処理に進み、これ以外のときには S114 の処理に進む。

ステップ 208 (S208) において、UI・AP 層実行管理部 480

(図10)は、データベース(DBMS302およびデータベース部300;図9)に対する操作のための第2の処理(DB操作#2)を開始し、データベース層のプログラムの処理を起動する。

第2の処理のためのデータベース層のプログラムは、第1の処理のためのプログラムと同様に、ライブラリ488により、コンポーネント化されたプログラムモジュールとして提供される。

ステップ110(S110)において、DB層実行管理部482(図10)は、データベース(DBMS302およびデータベース部300;図9)からの戻り値を判断し、S204において実行された処理が成功したか否かを判断する。

プログラム実行制御部46は、処理が成功したときにはS112の処理に進み、これ以外のときにはS114の処理に進む。

ステップ112(S112)において、DB層実行管理部482(図10)は、データベース(DBMS302およびデータベース部300;図9)に対する操作内容を確定(コミット)させ、処理結果および戻り値を、UI・AP層実行管理部480に返す。

ステップ114(S114)において、DB層実行管理部482は、データベースに対する操作内容を取消して、その内容を復元(ロールバック)し、処理結果および戻り値を、UI・AP層実行管理部480に返す。

ステップ116(S116)において、DB層実行管理部482は、データ管理部484を介して、データベースに対する接続を切り、データベースをクローズする。

UI・AP層実行管理部480は、DB層実行管理部482から入力された処理結果および戻り値に応じて、UI層制御部460およびAP層制御部462を介して、サービス提供部304を制御し、クライアントPC102(図7)の入力・出力装置126(図8)に、処理結果を表示させるなど、必要な処理を行わせる。

また、メモリ領域管理部486は、メモリ領域を解放し、トランザクション管理部490は、トランザクションの終了のための処理を行う。

なお、図 2 に例示したプログラムの処理においては、データベースの切断およびクローズが、個々のデータベース層の処理が成功したか否かにかかわらず、全てのデータベース層の処理が終了した後に、初めて行われる。

これに対して、図 1 1 ～図 1 3 に例示した制御プログラム 4 4 の処理は、  
5     、図 2 に例示した処理と比べて、データベース層の処理が最初に失敗したとき、  
      または、全てのデータベース層の処理が終了したときに行われる点で異なっている。

図 1 5 は、開発・制御用プログラム 4 0 （図 1 0 ）による制御プログラム 4 4 に対するデバッグの容易さを説明する図である。

10     図 1 5 に示すように、開発・制御用プログラム 4 0 （図 1 0 ）により制御プログラム 4 4 の開発およびそのデバッグを行うと、ユーザインターフェース層のプロセス、および、アプリケーション層およびデータベース層のプロセスの両方が、同じプログラム実行制御部 4 6 の制御の下で実行される。

従って、開発・制御用プログラム 4 0 を用いると、制御プログラム 4 4  
15     のユーザインターフェース層と、アプリケーション層およびデータベース層とで、  
      同じデバッガ 4 2 2 を用いてデバッグを行うことができる。

つまり、開発・制御用プログラム 4 0 を用いると、図 6 に示したように、ユーザインターフェース層と、アプリケーション層およびデータベース層とで、異なる複数のデバッガを用いてデバッグを行う必要がないので、図 6 に示した場合に比べて、制御プログラム 4 4 のデバッグが容易になる。  
20

図 1 6 は、各階層に複数のトランザクションを含む制御プログラム 4 4 を例示する図である。

なお、図 1 1 ～図 1 3 には、ある 1 つの階層において、1 つのトランザクションが定義されている制御プログラム 4 4 を例示したが、実際には、図 1 6  
25     に示すように、制御プログラム 4 4 の 1 つの階層において、複数のトランザクション（例えば、階層 # 1 における T 1 - 1 ～T 1 - 3 ）が定義されていてもよい。

また、あるトランザクション（例えば、T 1 - 1 ）により起動される処理において、さらにトランザクション（例えば、T 2 - 1 ～T 2 - 3 ）が定義さ

れていてもよい。

プログラム実行制御部 46（図 10）は、図 16 に例示した制御プログラム 44 に対しても、図 11～図 13 に例示した制御プログラム 44 に対してと同様な実行管理を行うことができる。

5

#### 産業上の利用可能性

本発明は、データベースの操作に利用可能である。

## 請求の範囲

1. 複数の階層に分かれた処理により、データベースを操作するデータベース操作装置であって、前記処理は、他の階層の処理を起動する起動処理、  
5 および、前記起動処理により起動される被起動処理またはこれらのいずれかであって、前記被起動処理の1つ以上は、前記データベースに対する操作を行うデータベース操作処理であって、

1つ以上の前記起動処理を含む処理集合を定義する集合定義手段と、  
前記処理集合に含まれる起動処理の進行と、前記処理集合に含まれる起動処理  
10 により起動されたデータベース操作処理の処理結果とに基づいて、少なくとも、このデータベース操作処理による前記データベースに対する操作の内容を制御する処理制御手段と  
を有するデータベース操作装置。

15 2. 前記階層は、3つ以上に分かれ、  
前記処理制御手段は、前記処理集合に含まれる起動処理の進行と、前記処理集合に含まれる起動処理により起動されたデータベース操作処理以外の被起動処理の処理結果とに応じて、前記データベース操作処理以外の被起動処理の処理内容を、さらに制御する  
20 1に記載のデータベース操作装置。

3. 1つ以上の前記データベース操作処理を含むライブラリ手段  
をさらに有し、  
前記データベース操作処理を起動する起動処理は、前記ライブラリ手段に含ま  
25 れる前記データベース操作処理を起動する  
1に記載のデータベース操作装置。

4. 前記処理集合それぞれに対して、この処理集合に含まれる起動処理、および、この処理集合に含まれる起動処理により起動される被起動処理のた

めの記憶領域を設定する記憶領域設定手段と、

前記処理集合それぞれに含まれる処理のために用いられるデータを、前記処理集合それぞれに対して設定された記憶領域において管理するデータ管理手段とをさらに有する 1 または 3 に記載のデータベース操作装置。

5

5. 前記被起動処理それぞれは、この被起動処理を起動した起動処理に対して、処理の結果を示す戻り値を返し、

前記階層は、

インターフェース層と、

10     アプリケーション層と、

データベース層と

を含み、

前記インターフェース層は、

15     前記起動処理として、外部からの操作に応じて、前記アプリケーション層に含まれる被起動処理を起動し、起動した被起動処理から返された戻り値に応じた処理を行うユーザインターフェース処理、1つ以上

を含み、

前記アプリケーション層は、

20     前記起動処理および前記被起動処理として、前記インターフェース処理により起動され、前記データベース層に含まれるデータベース操作処理を1つ以上、起動し、前記起動されたデータベース操作処理からの戻り値に基づいて、前記データベースを用いたサービスを実現し、前記戻り値として、このサービスの結果を、前記ユーザインターフェース処理に返すアプリケーション処理、1つ以上

を含み、

25     データベース層は、

前記データベース操作処理として、前記アプリケーション処理により起動され、前記データベースに対する操作を行い、前記戻り値として、このデータベースに対する操作の結果を、前記アプリケーション処理に返すデータベース操作処理、1つ以上

を含む

1, 3または4に記載のデータベース操作装置。

6. 前記処理制御手段は、

- 5 前記処理集合に含まれる前記アプリケーション処理により、前記データベース操作処理が、最初に起動されたときに、前記起動されたデータベース操作処理と、前記データベースとを接続し、

前記処理集合に含まれる前記アプリケーション処理により、最後に起動された前記データベース操作処理が終了したとき、または、前記データベース操作処理  
10 が失敗したときに、前記起動されたデータベース操作処理と、前記データベースとを切断する

ように、前記データベース操作処理の実行を制御する

5に記載のデータベース操作装置。

- 15 7. 前記処理制御手段は、

前記処理集合に含まれる前記アプリケーション処理により、最後に起動された前記データベース操作処理が成功したときに、前記処理集合に含まれる前記アプリケーション処理により起動された前記データベース操作処理の結果を、前記データベースにおいて確定させ、

- 20 これ以外のときには、前記データベースを、前記処理集合に含まれる前記アプリケーション処理により、前記データベース操作処理が最初に起動される前の状態に戻す

5または6に記載のデータベース操作装置。

- 25 8. 前記処理制御手段は、

前記処理集合に含まれる前記アプリケーション処理により起動された前記データベース操作処理が失敗したときに、前記データベースを、前記処理集合に含まれる前記アプリケーション処理により、前記データベース操作処理が最初に起動される前の状態に戻す

5 ～ 7 のいずれかに記載のデータベース操作装置。

9. 複数の階層に分かれた処理により、所定の情報処理を行う情報処理装置であって、前記処理は、他の階層の処理を起動する起動処理、および、前記起動処理により起動される被起動処理またはこれらのいずれかであって、
- 5 1つ以上の前記起動処理を含む処理集合を定義する集合定義手段と、
- 前記処理集合に含まれる起動処理の進行と、前記処理集合に含まれる起動処理により起動された被起動処理の処理結果とに応じて、この被起動処理の処理内容を制御する処理内容制御手段と
- 10 を有する情報処理装置。

10. 複数の階層に分かれた処理により、データベースを操作するデータベース操作方法であって、前記処理は、他の階層の処理を起動する起動処理、および、前記起動処理により起動される被起動処理またはこれらのいずれかであって、前記被起動処理の1つ以上は、前記データベースに対する操作を行うデータベース操作処理であって、
- 15 1つ以上の前記起動処理を含む処理集合を定義する定義ステップと、
- 前記処理集合に含まれる起動処理の進行と、前記処理集合に含まれる起動処理により起動されたデータベース操作処理の処理結果とに基づいて、少なくとも、
- 20 このデータベース操作処理による前記データベースに対する操作の内容を制御する処理制御ステップと
- を含むデータベース操作方法。

11. 前記階層は、3つ以上に分かれ、
- 25 処理制御ステップは、前記処理集合に含まれる起動処理の進行と、前記処理集合に含まれる起動処理により起動されたデータベース操作処理以外の被起動処理の処理結果とに応じて、前記データベース操作処理以外の被起動処理の処理内容を、さらに制御する

10に記載のデータベース操作方法。

1 2. 複数の階層に分かれた処理により、所定の情報処理を行う情報処理方法であって、前記処理は、他の階層の処理を起動する起動処理、および、前記起動処理により起動される被起動処理またはこれらのいずれかであって、

- 5 1つ以上の前記起動処理を含む処理集合を定義する集合定義ステップと、  
前記処理集合に含まれる起動処理の進行と、前記処理集合に含まれる起動処理により起動された被起動処理の処理結果とに応じて、この被起動処理の処理内容を制御する処理内容制御ステップと  
を含む情報処理方法。

10

1 3. 複数の階層に分かれた処理により、データベースを操作するデータベース操作装置のプログラムであって、前記処理は、他の階層の処理を起動する起動処理、および、前記起動処理により起動される被起動処理またはこれらのいずれかであって、前記被起動処理の1つ以上は、前記データベースに対する  
15 操作を行うデータベース操作処理であって、

- 1つ以上の前記起動処理を含む処理集合を定義する集合定義ステップと、  
前記処理集合に含まれる起動処理の進行と、前記処理集合に含まれる起動処理により起動されたデータベース操作処理の処理結果とに基づいて、少なくとも、このデータベース操作処理による前記データベースに対する操作の内容を制御する  
20 処理制御ステップと  
をコンピュータに実行させるプログラム。

1 4. 前記階層は、3つ以上に分かれ、

- 前記処理制御ステップは、前記処理集合に含まれる起動処理の進行と、前記処  
25 理集合に含まれる起動処理により起動されたデータベース操作処理以外の被起動処理の処理結果とに応じて、前記データベース操作処理以外の被起動処理の処理内容を、さらに制御する

1 3に記載のプログラム。

15. 1つ以上の前記データベース操作処理を含むライブラリをさらに有し、  
前記データベース操作処理を起動する起動処理は、前記ライブラリに含まれる前記データベース操作処理を起動する
- 5 13に記載のプログラム。
16. 前記処理集合それぞれに対して、この処理集合に含まれる起動処理、および、この処理集合に含まれる起動処理により起動される被起動処理のための記憶領域を設定する記憶領域設定ステップと、
- 10 前記処理集合それぞれに含まれる処理のために用いられるデータを、前記処理集合それぞれに対して設定された記憶領域において管理するデータ管理ステップと
- をさらにコンピュータに実行させる13または15に記載のプログラム。
17. 前記被起動処理それぞれは、この被起動処理を起動した起動処理に対して、処理の結果を示す戻り値を返し、  
前記階層は、  
インターフェース層と、  
アプリケーション層と、  
20 データベース層と  
を含み、  
前記インターフェース層は、  
前記起動処理として、外部からの操作に応じて、前記アプリケーション層に含まれる被起動処理を起動し、起動した被起動処理から返された戻り値に応じた処理を行うユーザインターフェース処理、1つ以上  
25 を含み、  
前記アプリケーション層は、  
前記起動処理および前記被起動処理として、前記インターフェース処理により起動され、前記データベース層に含まれるデータベース操作処理を1つ以上、起

動し、前記起動されたデータベース操作処理からの戻り値に基づいて、前記データベースを用いたサービスを実現し、前記戻り値として、このサービスの結果を、前記ユーザインターフェース処理に返すアプリケーション処理、1つ以上を含み、

5 データベース層は、

前記データベース操作処理として、前記アプリケーション処理により起動され、前記データベースに対する操作を行い、前記戻り値として、このデータベースに対する操作の結果を、前記アプリケーション処理に返すデータベース操作処理、1つ以上

10 を含む

13, 15または16に記載のプログラム。

18. 前記処理制御ステップは、

前記処理集合に含まれる前記アプリケーション処理により、前記データベース  
15 操作処理が、最初に起動されたときに、前記起動されたデータベース操作処理と、前記データベースとを接続し、

前記処理集合に含まれる前記アプリケーション処理により、最後に起動された前記データベース操作処理が終了したとき、または、前記データベース操作処理が失敗したときに、前記起動されたデータベース操作処理と、前記データベース  
20 とを切断する

ように、前記データベース操作処理の実行を制御する

17に記載のプログラム。

19. 前記処理制御ステップは、

前記処理集合に含まれる前記アプリケーション処理により、最後に起動された前記データベース操作処理が成功したときに、前記処理集合に含まれる前記アプリケーション処理により起動された前記データベース操作処理の結果を、前記データベースにおいて確定させ、

これ以外のときには、前記データベースを、前記処理集合に含まれる前記アプ

リケーション処理により、前記データベース操作処理が最初に起動される前の状態に戻す

17または18に記載のプログラム。

5           20.   前記処理制御ステップは、

前記処理集合に含まれる前記アプリケーション処理により起動された前記データベース操作処理が失敗したときに、前記データベースを、前記処理集合に含まれる前記アプリケーション処理により、前記データベース操作処理が最初に起動される前の状態に戻す

10          17～19のいずれかに記載のプログラム。

21.   複数の階層に分かれた処理により、所定の情報処理を行う情報処理装置のプログラムであって、前記処理は、他の階層の処理を起動する起動処理、および、前記起動処理により起動される被起動処理またはこれらのいずれか

15   であって、

1つ以上の前記起動処理を含む処理集合を定義する集合定義ステップと、

前記処理集合に含まれる起動処理の進行と、前記処理集合に含まれる起動処理により起動された被起動処理の処理結果とに応じて、この被起動処理の処理内容を制御する処理内容制御ステップと

20          をコンピュータに実行させるプログラム。

## 図 1

```
private void button5_Click()
{
    // DB への接続を行い、トランザクションを開始する
    DBConnection con = new DBConnection(接続情報);
    con.Open();
    DBTransaction tx = con.BeginTransaction();

    // DB への操作を行う
    AuthorsDataSet ds = new AuthorsDataSet();
    AuthorsDataSet.authorsRow row = ds.authors.NewauthorsRow();
    ...
    ds.authors.AddauthorsRow(row);
    AuthorsAdapter adp = new AuthorsAdapter();
    adp.Update(con, tx, ds.authors);

    // 次の DB 操作を行う
    ...

    // 通常はこのように複数の DB 操作を行う
    ...

    // トランザクションをコミットし、DB 接続を終了する
    tx.Commit();
    con.Close();
}
```

## 図 2

## UI 層のプログラム

```
private void button5_Click()
{
    App.PubsApp app = new App.PubsApp();
    app.InsertAuthorAndStore(textBox1.Text, textBox2.Text);
}
```

## AP 層のプログラム

```
public bool InsertAuthorAndStore(string au_id, string store_id)
{
    // DB への接続を行い、トランザクションを開始する
    DBConnection con = new DBConnection(接続情報);
    con.Open();
    DBTransaction tx = con.BeginTransaction();

    // DB 操作 # 1
    DB.Data.AuthorsData data1 = new DB.Data.AuthorsData();
    bool ret1 = data1.Insert(con, tx, au_id);

    // DB 操作 # 2
    DB.Data.StoresData data2 = new DB.Data.StoresData();
    bool ret2 = data2.Insert(con, tx, store_id);

    // 全ての処理が成功したらトランザクションをコミットし DB 接続を終了する
    tx.Commit();
    con.Close();

    return true;
}
```

## DB 層のプログラム

```
public bool Insert(DBConnection con, DBTransaction tx, string au_id)
{
    DB.AuthorsDB db = new DB.AuthorsDB();
    bool ret = db.Insert(con, tx, au_id);
    ...
    return ret;
}
```

図 3

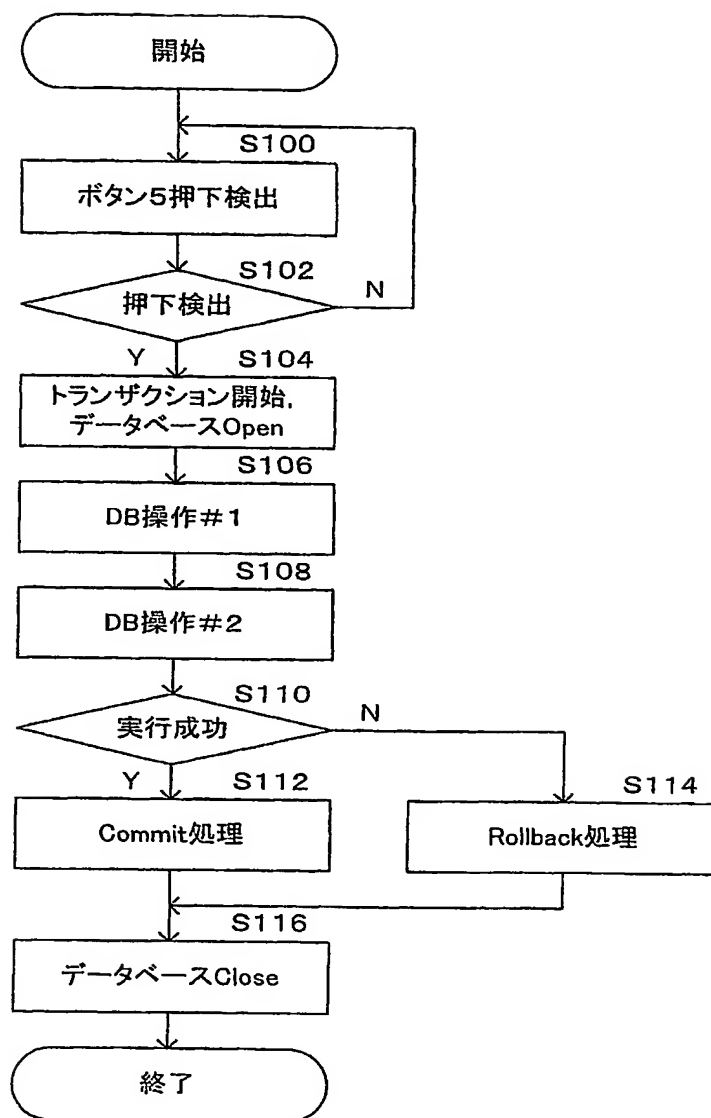
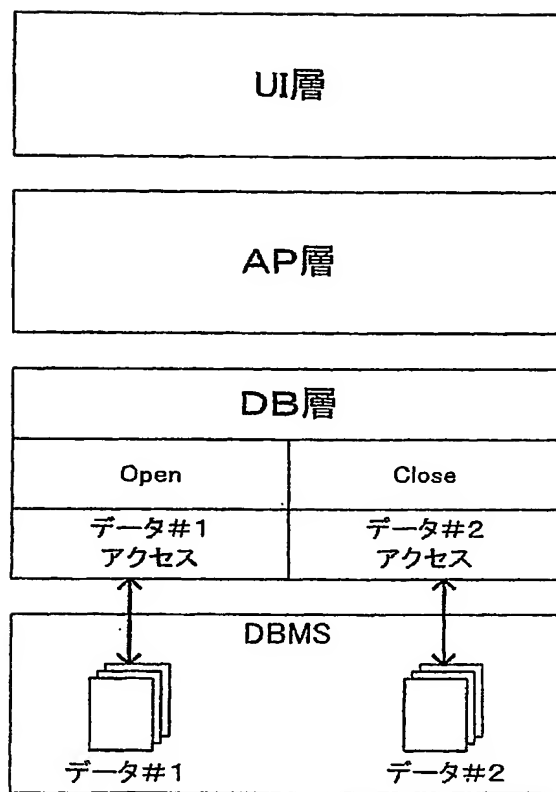
S10

図 4



## 図 5

## UI 層のプログラム

```
private void button5_Click ()
{
    App.PubsApp app = new App.PubsApp ();
    app.InsertAuthorAndStore(textBox1.Text, textBox2.Text);
}
```

## AP 層のプログラム

[トランザクションの動作を指定する属性を指定]

```
public bool InsertAuthorAndStore(string au_id, string store_id)
{
    // DB 操作 # 1
    DB.Data.AuthorsData data1 = new DB.Data.AuthorsData ();
    bool ret1 = data1.Insert(con, tx, au_id);

    // DB 操作 # 2
    DB.Data.StoresData data2 = new DB.Data.StoresData ();
    bool ret2 = data2.Insert(con, tx, store_id);

    return true;
}
```

## DB 層のプログラム

```
public bool Insert(string au_id)
{
    SqlConnection con = new SqlConnection(strCon);
    con.Open();
    DB.AuthorsDB db = new DB.AuthorsDB ();
    bool ret = db.Insert(con, tx, au_id);
    if (ret)
        ContextUtil.SetCommit ();
    else
        ContextUtil.SetAbort ();

    con.Close();
    return ret;
}
```

図 6

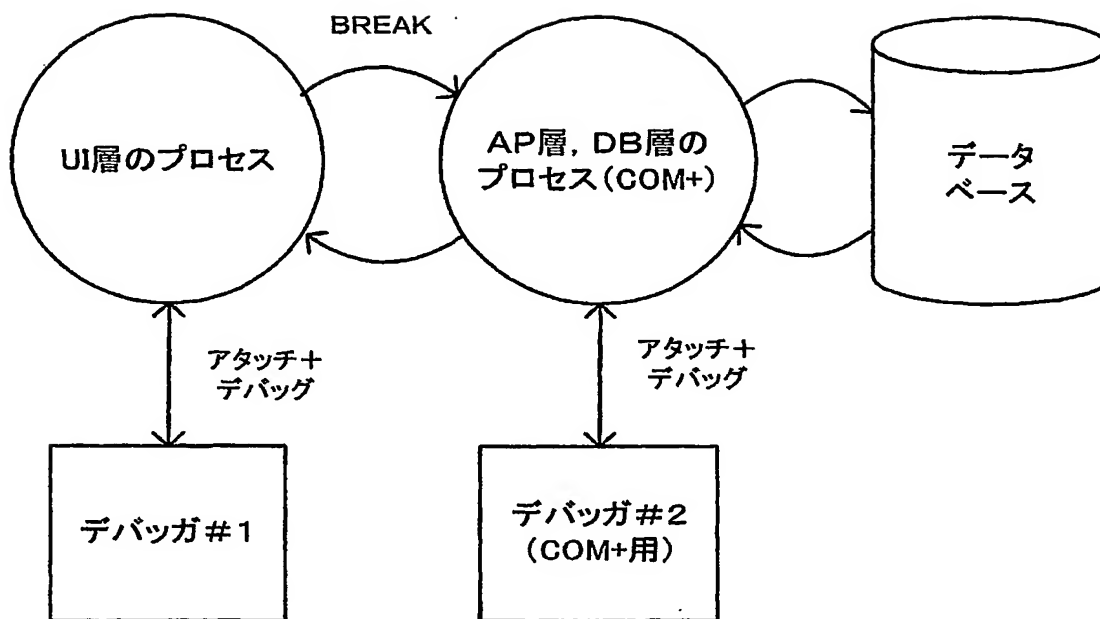


図 7

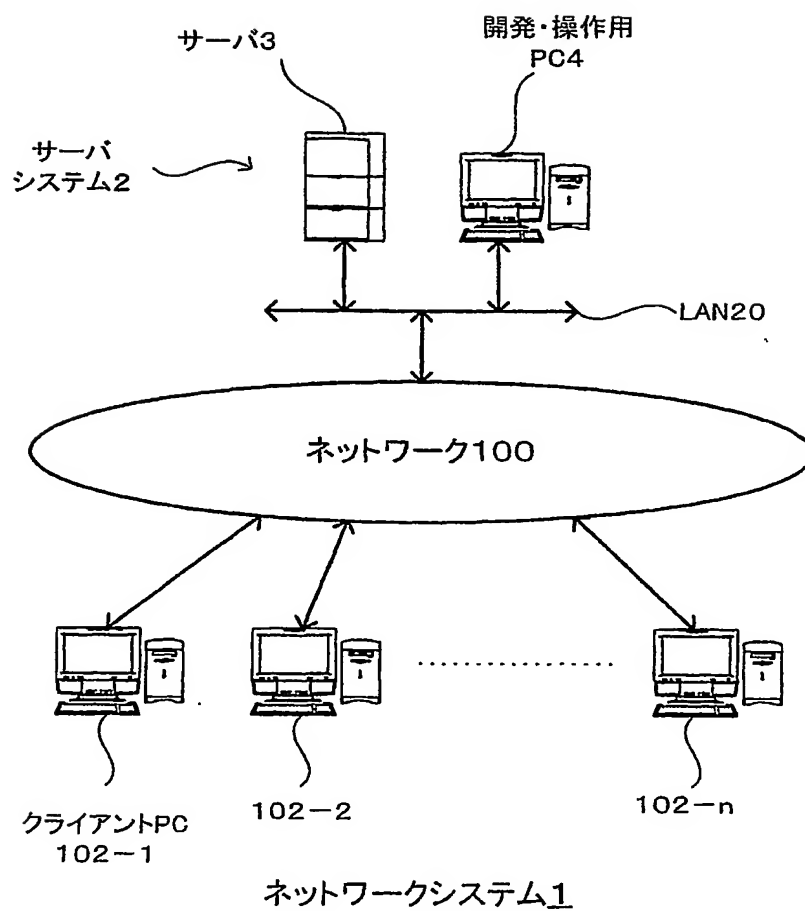
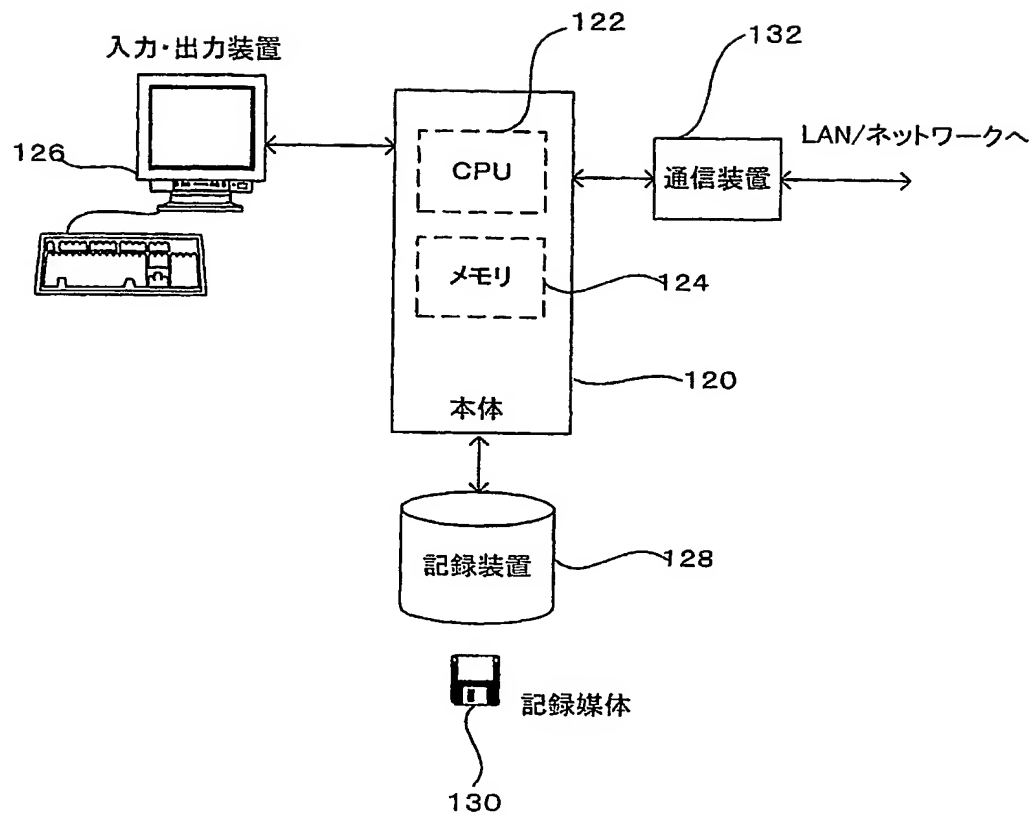
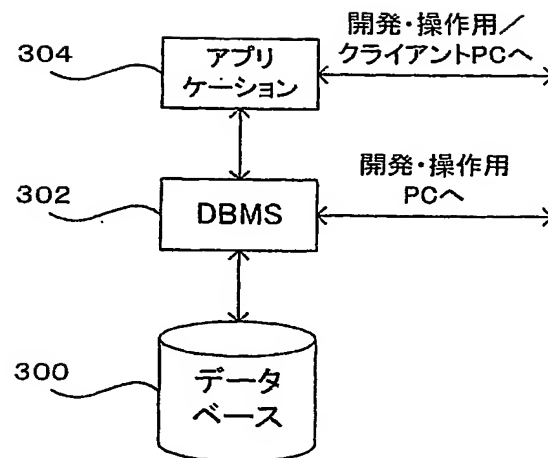


図 8



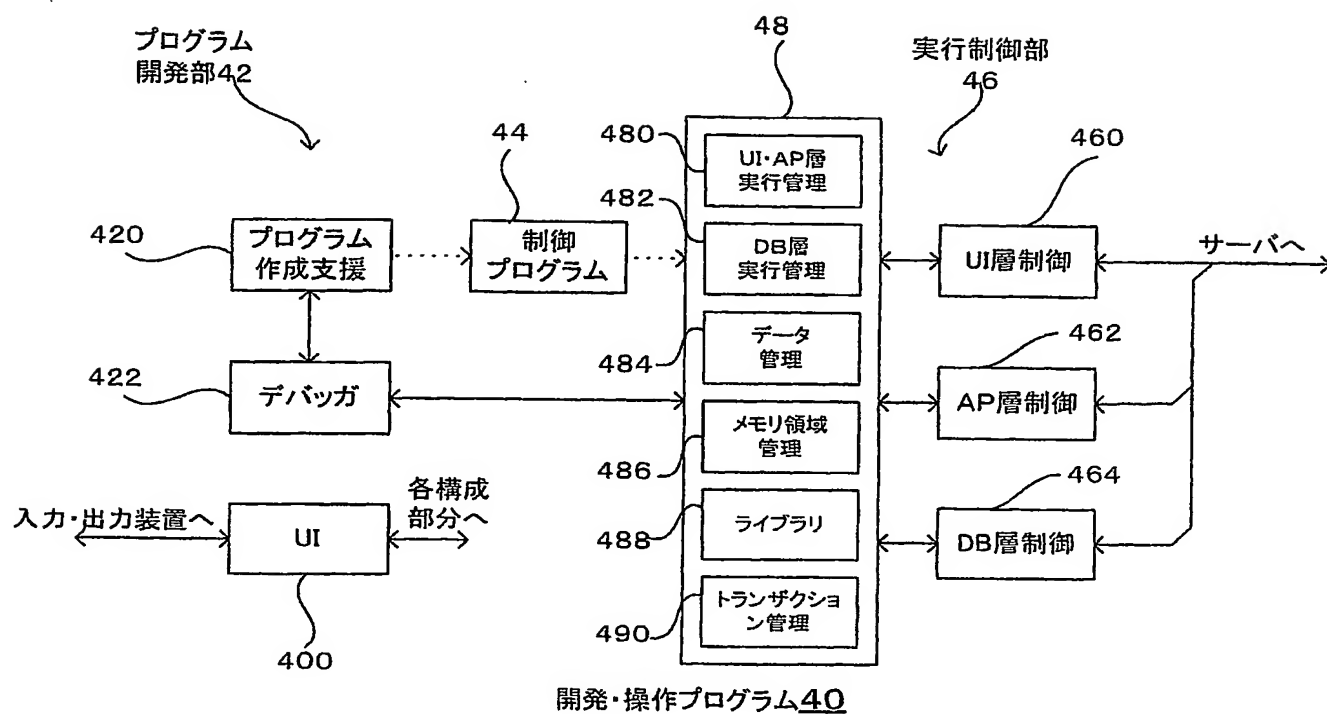
サーバ3,  
PC4, 102

図 9



サーバプログラム30

図 10



## 図 1 1

## UI 層のプログラム

```
private void button5_Click()
{
    App.PubsApp app = new App.PubsApp();
    app.InsertAuthorAndStore(textBox1.Text, textBox2.Text);
}
```

## AP 層のプログラム

[トランザクションの動作を指定する属性を指定]

```
public bool InsertAuthorAndStore(string au_id, string store_id)
{
    // トランザクションの定義
    using(new TransactionUnit())
    {
        // DB 操作 # 1
        DB.Data.AuthorsData data1 = new DB.Data.AuthorsData();
        bool ret1 = data1.Insert(con, tx, au_id);

        // DB 操作 # 2
        DB.Data.StoresData data2 = new DB.Data.StoresData();
        bool ret2 = data2.Insert(con, tx, store_id);
    }
    return true;
}
```

## DB 層のプログラム

```
public bool Insert(string au_id)
{
    MySqlConnection con = new MySqlConnection(strCon);
    con.Open();
    MyTransaction tx = con.BeginTransaction();
    DB.AuthorsDB db = new DB.AuthorsDB();
    bool ret = db.Insert(con, tx, au_id);
    if (ret)
        tx.Commit();
    else
        tx.Rollback();

    con.Close();
    return ret;
}
```

図 1 2

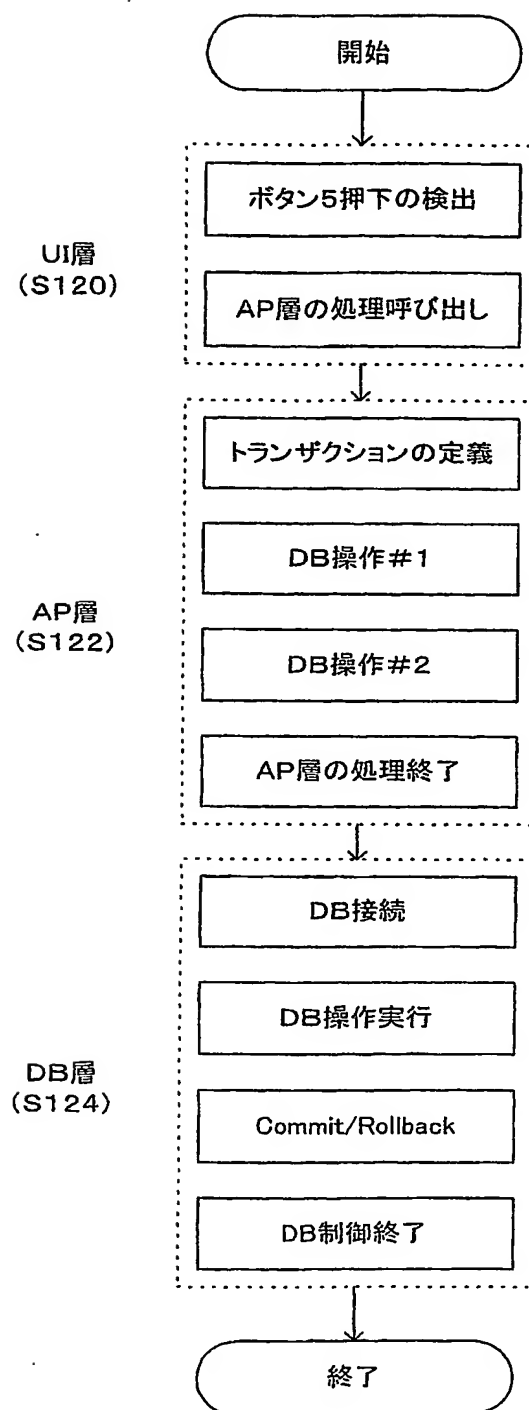


図 1 3

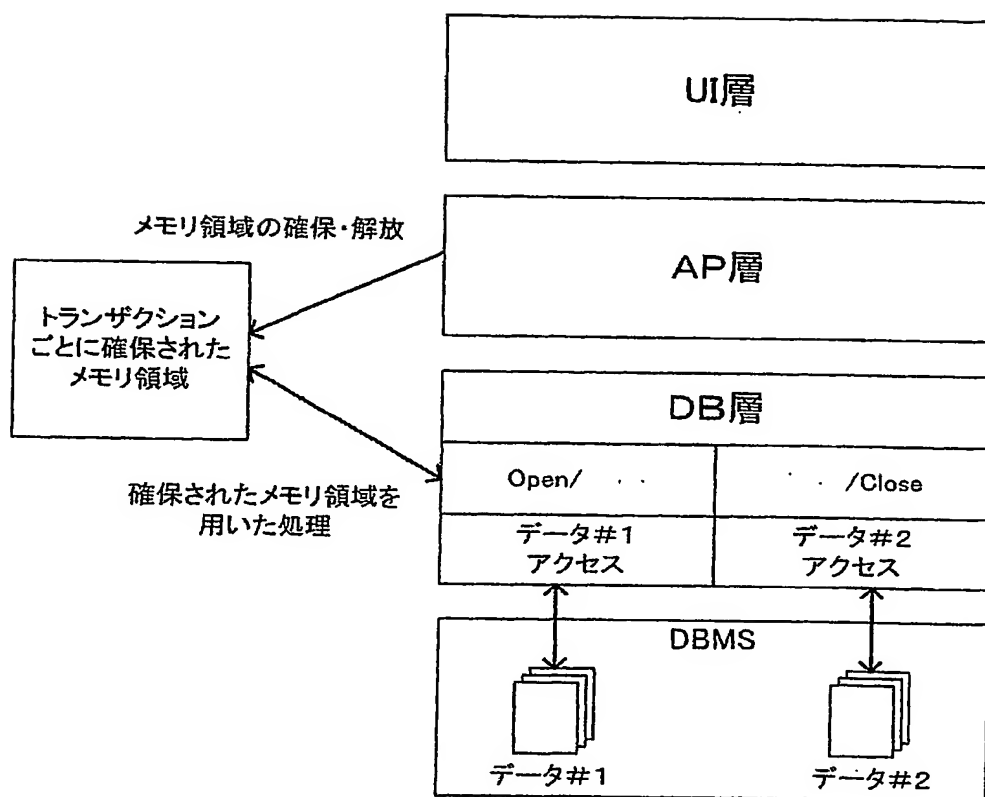


図 1 4

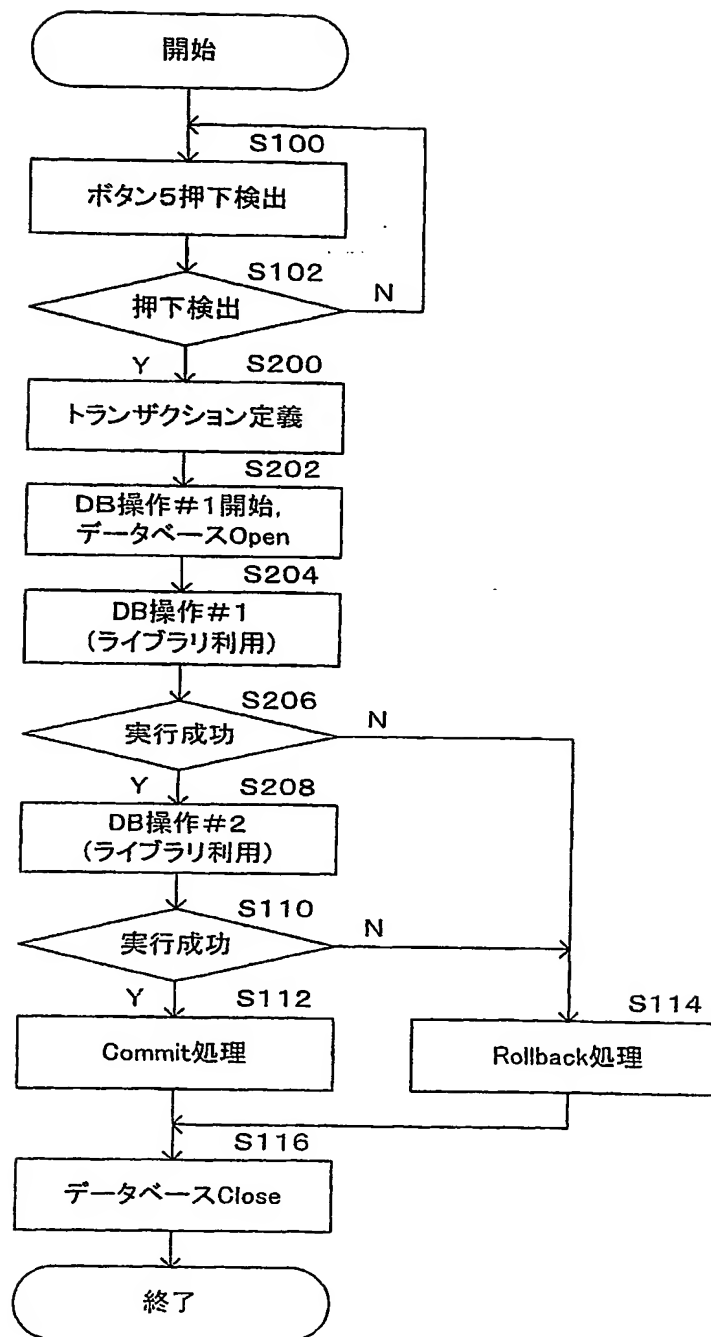
S20

図 15

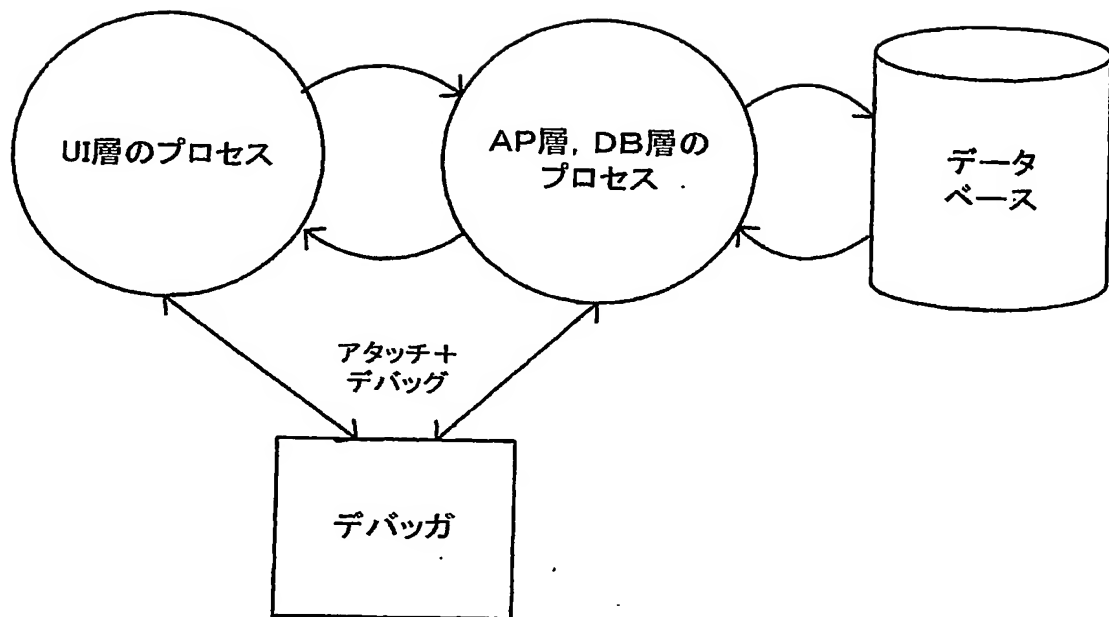
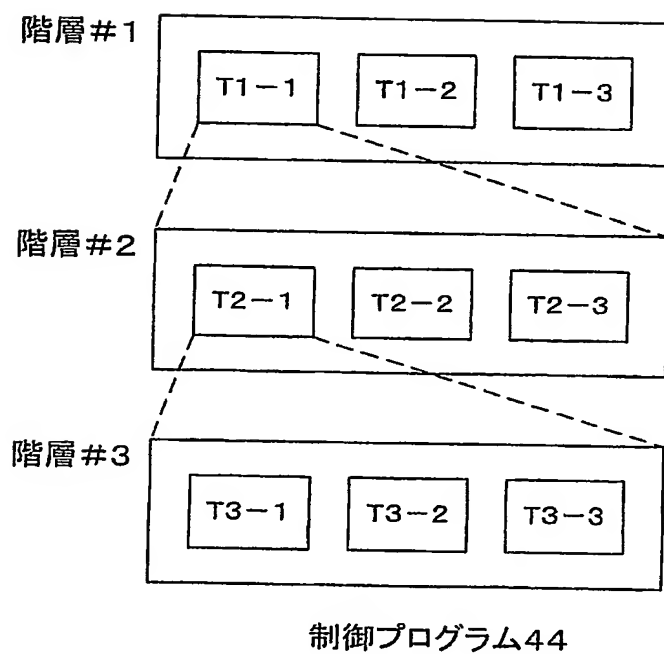


図 16



# INTERNATIONAL SEARCH REPORT

International application No.

PCT/JP2004/000521

## A. CLASSIFICATION OF SUBJECT MATTER

Int.Cl<sup>7</sup> G06F12/00

According to International Patent Classification (IPC) or to both national classification and IPC

## B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

Int.Cl<sup>7</sup> G06F12/00

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Jitsuyo Shinan Koho	1922-1996	Jitsuyo Shinan Toroku Koho	1996-2004
Kokai Jitsuyo Shinan Koho	1971-2004	Toroku Jitsuyo Shinan Koho	1994-2004

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

## C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	JP 2003-140922 A (NEC Corp.), 16 May, 2003 (16.05.03), Full text; Figs. 1 to 3 (Family: none)	1-3, 5-15, 17-21
Y	JP 61-188640 A (Kokusai Denshin Denwa Co., Ltd. (KDD)), 22 August, 1986 (22.08.86), Full text; Figs. 1 to 15 (Family: none)	1-3, 5-15, 17-21
Y A	JP 11-338746 A (NTT Communicationware Corp.), 10 December, 1999 (10.12.99), Full text; Figs. 1 to 7 & US 006374243 B1	5-8, 17-20 4, 16



Further documents are listed in the continuation of Box C.



See patent family annex.

* Special categories of cited documents:	"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"A" document defining the general state of the art which is not considered to be of particular relevance	"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"E" earlier document but published on or after the international filing date	"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)	"&" document member of the same patent family
"O" document referring to an oral disclosure, use, exhibition or other means	
"P" document published prior to the international filing date but later than the priority date claimed	

Date of the actual completion of the international search  
16 February, 2004 (16.02.04)

Date of mailing of the international search report  
02 March, 2004 (02.03.04)

Name and mailing address of the ISA/  
Japanese Patent Office

Authorized officer

Facsimile No.

Telephone No.

## A. 発明の属する分野の分類 (国際特許分類 (IPC))

Int. Cl<sup>7</sup> G06F12/00

## B. 調査を行った分野

調査を行った最小限資料 (国際特許分類 (IPC))

Int. Cl<sup>7</sup> G06F12/00

最小限資料以外の資料で調査を行った分野に含まれるもの

日本国実用新案公報 1922-1996年

日本国公開実用新案公報 1971-2004年

日本国実用新案登録公報 1996-2004年

日本国登録実用新案公報 1994-2004年

国際調査で使用した電子データベース (データベースの名称、調査に使用した用語)

## C. 関連すると認められる文献

引用文献の カテゴリー*	引用文献名 及び一部の箇所が関連するときは、その関連する箇所の表示	関連する 請求の範囲の番号
Y	JP 2003-140922 A (日本電気株式会社) 2003.05.16, 全文, 第1-3図 (ファミリーなし)	1-3, 5-15, 17-21
Y	JP 61-188640 A (国際電信電話株式会社) 1986.08.22, 全文, 第1-15図 (ファミリーなし)	1-3, 5-15, 17-21
Y	JP 11-338746 A (エヌ・ティ・ティ・コミュニケーションウェア株式会社) 1999.12.10, 全文, 第1-7図 & US 006374243 B1	5-8, 17-20
A		4, 16

☐ C欄の続きにも文献が列挙されている。☐ パテントファミリーに関する別紙を参照。

## \* 引用文献のカテゴリー

「A」 特に関連のある文献ではなく、一般的技術水準を示すもの

「E」 国際出願日前の出願または特許であるが、国際出願日以後に公表されたもの

「L」 優先権主張に疑義を提起する文献又は他の文献の発行日若しくは他の特別な理由を確立するために引用する文献 (理由を付す)

「O」 口頭による開示、使用、展示等に言及する文献

「P」 国際出願日前で、かつ優先権の主張の基礎となる出願

の日の後に公表された文献

「T」 国際出願日又は優先日後に公表された文献であって出願と矛盾するものではなく、発明の原理又は理論の理解のために引用するもの

「X」 特に関連のある文献であって、当該文献のみで発明の新規性又は進歩性がないと考えられるもの

「Y」 特に関連のある文献であって、当該文献と他の1以上の文献との、当業者にとって自明である組合せによって進歩性がないと考えられるもの

「&amp;」 同一パテントファミリー文献

国際調査を完了した日

16.02.2004

国際調査報告の発送日

02.3.2004

国際調査機関の名称及びあて先

日本国特許庁 (ISA/JP)

郵便番号100-8915

東京都千代田区霞が関三丁目4番3号

特許庁審査官 (権限のある職員)

土田 行一

5N

9751

電話番号 03-3581-1101 内線 3545